Pseudocode Introduction: Iteration

Official Cambridge Syntax Compliant Pack

Teacher	Avik Ghosh	**Subject**	Computing
Unit	Computational Thinking	**Duration**	40 Minutes

🎯 Curriculum Standards & Success Criteria

- **9CT.01:** Follow, understand, edit and correct algorithms that are presented as pseudocode.
- **9CT.02:** Follow flowchart or pseudocode algorithms that use loops.
- **Syntax Rules Applied:** Variable initialization via `DECLARE`, explicit assignment arrows (`←`), exact loop boundaries, and `<>` inequality checks.

PART 1: Student Reference Notes

1. Introduction to Iteration

In computer science, **iteration** means the structured repetition of a block of statements. Instead of rewriting identical steps manually, loops enable automation, reduce code footprint, and ensure execution efficiency.

2. Standard Loop Structures

A. Count-Controlled Loops (`FOR` Loop)

Use a count-controlled loop when you know **exactly** how many times the loop block must run before execution begins.

Cambridge Syntax Requirements:

- Variables must be declared with `DECLARE` before use.
- The assignment operator `←` sets initial loop limits.
- The `NEXT` keyword must explicitly name the counter identifier.

```
DECLARE Counter : INTEGER
FOR Counter ← 1 TO 10
```

```
    OUTPUT Counter
NEXT Counter
```

B. Condition-Controlled Loops (`WHILE` Loop)

Use a condition-controlled loop when the total number of loops is **unknown** beforehand. The block runs continuously *while* a logical test is true.

```
DECLARE PasswordText : STRING
INPUT PasswordText
WHILE PasswordText <> "Secret123"
    OUTPUT "Access Denied. Enter your password:"
    INPUT PasswordText
ENDWHILE
```

Note: Under standard Cambridge conventions, the inequality operator is written as `<>` rather than `!=`.

3. Dry-Run Tracing Tables

A **Trace Table** records the progressive values of variables row-by-row during loop cycles to safely catch logic flaws.

Sample Algorithm:

```
DECLARE TotalSum : INTEGER
DECLARE LoopIndex : INTEGER

TotalSum ← 0
FOR LoopIndex ← 1 TO 3
    TotalSum ← TotalSum + LoopIndex
NEXT LoopIndex
```

Loop Iteration	Variable `LoopIndex`	Calculation Step	Variable `TotalSum`
Initial State	-	-	0
Iteration 1	1	0 + 1	1
Iteration 2	2	1 + 2	3
Iteration 3	3	3 + 3	**6**

PART 2: Classwork Worksheet

Topic: The Code Medic & Trace Drills

Student Name: ---------- **Date:** ----------

TASK 1: GUIDED PRACTICE

Variable Dry-Run Trace Table

Complete the missing trace elements for this algorithm working in pairs:

```
DECLARE MultiplierValue : INTEGER
DECLARE LoopCount : INTEGER
DECLARE CalculationResult : INTEGER

MultiplierValue ← 2
FOR LoopCount ← 1 TO 4
    CalculationResult ← LoopCount * MultiplierValue
NEXT LoopCount
```

Iteration	Variable MultiplierValue	Variable LoopCount	Variable CalculationResult
Initial State	2	-	-
1	2	1	[Fill in]
2	2	2	
3	2	3	
4	2	4	

TASK 2: INDEPENDENT PRACTICE

The Code Medic - Debugging Drills

Problem A: The Forever Trap

The code block below contains a logic defect making it loop infinitely.

```
DECLARE LoopX : INTEGER
LoopX ← 1
WHILE LoopX < 6
    OUTPUT LoopX
ENDWHILE
```

1. Identify and explain the logic flaw:

2. Rewrite the corrected algorithm using proper Cambridge syntax:

Problem B: The Skeleton Loop Blank Fill

Fill in the correct keywords below to iterate safely through all **even numbers** from 2 to 10.

```
DECLARE StepCounter : INTEGER

_________ StepCounter ← 2 TO 10 _________2
    OUTPUT ____________________
_________ StepCounter
```

TASK 3: PLENARY REFLECTION

Loop Selection Analysis

Scenario: You are creating a game profile loop. It must ask for a PIN until correct. However, if a counter ticks to 3 wrong attempts, it locks the game completely.

1. Which structural loop type is best suited for the main conditional repetition?

2. Justify your architectural choice:

TASK 4: HOMEWORK TASK

Rocket Launch Countdown Challenge

Prompt: Write a valid pseudocode block counting backwards from 10 down to 1, followed by outputting `"Blast Off!"`.

Hint: Think about your assignment loops or negative STEP value changes!